Trajectory Improvement and Reward Learning from Comparative Language Feedback

Zhaojing Yang[1] **Miru Jun**[1] **Jeremy Tien**[2]
Stuart J. Russell[2] **Anca Dragan**[2] **Erdem Bıyık**[1]

[1]University of Southern California [2]University of California, Berkeley

Abstract: Learning from human feedback has gained traction in fields like robotics and natural language processing in recent years. While prior works mostly rely on human feedback in the form of comparisons, language is a preferable modality that provides more informative insights into user preferences. In this work, we aim to incorporate comparative language feedback to iteratively improve robot trajectories and to learn reward functions that encode human preferences. To achieve this goal, we learn a shared latent space that integrates trajectory data and language feedback, and subsequently leverage the learned latent space to improve trajectories and learn human preferences. To the best of our knowledge, we are the first to incorporate comparative language feedback into reward learning. Our simulation experiments demonstrate the effectiveness of the learned latent space and the success of our learning algorithms. We also conduct human subject studies that show our reward learning algorithm achieves a 23.9% higher subjective score on average and is 11.3% more time-efficient compared to preference-based reward learning, underscoring the superior performance of our method. Our website is at https://liralab.usc.edu/comparative-language-feedback/.

Keywords: Learning from human feedback, reward learning, HRI

1 Introduction

Learning from human feedback has gained significant popularity in robotics, leading to the study of different forms of human feedback: demonstrations [1, 2, 3, 4], preference comparisons [5, 6, 7, 8], rankings [9], physical corrections [10], visual saliency maps [11], human language [12, 13], etc. Among these, preference comparisons grew popular for its simplicity and ease of use, especially compared to demonstrations [7]. Preference comparisons often involve users choosing between a pair of choices. Using these selections to learn a reward function and train a policy is known as reinforcement learning from human feedback (RLHF) [6] or more generally preference-based learning [5, 14]. It has proven applicable to a broad range of fields ranging from robotics [7, 15] to natural language processing [16], from traffic routing [17] to human-computer interaction [18].

Despite their successes, preference comparisons suffer from problems [19] such as the unreliability of human data and the limited information bandwidth, i.e., each pairwise comparison contains at most 1 bit of information [20]. There has been research to provide a better interface [21], allowing the users to specify their preferences for every feature, but they require features to be designed by hand. As an alternative form of human feedback, comparative language is considerably more informative than preference comparisons, allowing users to prioritize specific aspects. For example, it allows users to naturally indicate their preference about speed by simply saying, "the robot should move faster," making it more intuitive and interpretable.

In this work, we aim to leverage comparative language feedback to learn the human preferences, i.e., their reward functions. In pursuit of this objective, we first learn a shared latent space that aligns trajectories and language feedback. This alignment enables the robot to comprehend human language feedback, leverage it for adapting its behavior to learn and better align with the human's preferences. To test the effectiveness of our approach, we conduct experiments in two simulation environments and a human-subject study with a real robot. The results suggest that reward learning from

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

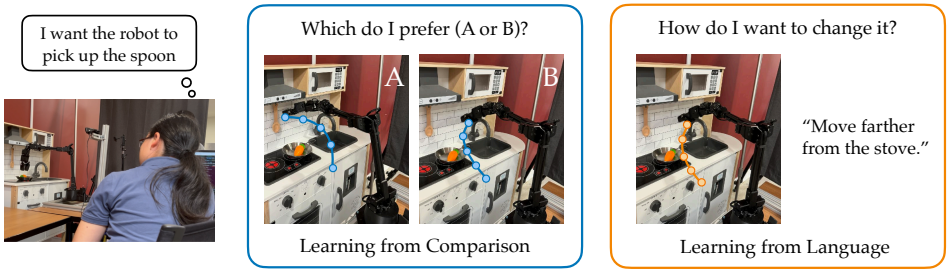


Figure 1: An image from our human subject studies where the human user wants the robot to pick up the spoon. Compared to traditional comparison preference learning, our language preference learning enables users to give more informative feedback, which helps the robot to capture human preferences more efficiently.

comparative language feedback outperforms traditional preference comparisons in performance and time-efficiency, and is favored by most of the users.

2 Related Work

Before formally defining the problem, we will first review existing works in learning from human feedback, preference-based learning, and robot learning from human language feedback.

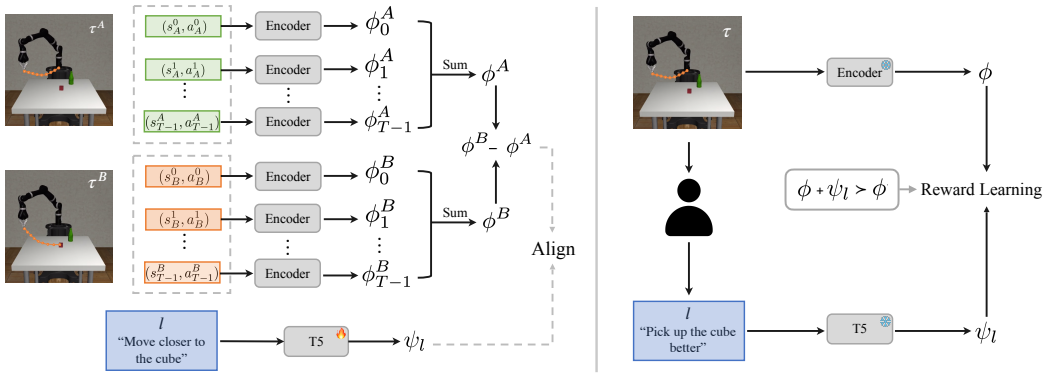
Learning from Human Feedback. Several promising approaches leverage human feedback to train robots [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 22]. However, the most popular methods of learning from demonstrations and rankings involve a trade-off between informativeness and ease of use [6, 19, 23, 24]. Furthermore, none has taken advantage of the expressive nature of language to develop a method of reward learning from comparative language feedback. Addressing this gap will provide more scalable methods for training robots, particularly in complex environments where traditional feedback forms may be insufficient or impractical.

Preference-based Learning. Based on preference comparison feedback, preference-based learning is widely used for its logical intuitiveness and ease of use. Learning from rankings is beneficial to deep reinforcement learning [6], but suffers from several shortcomings [19] such as the reliability of human data and difficulties of choosing from equivalent choices. Works have sought to improve its shortcomings in time- and sample-efficiency by actively generating pairs based on information gain [20] or volume removal [5], and doing these in batches of pairs [25]. Sikchi et al. [26] and Brown et al. [27] integrate preference feedback into imitation learning, demonstrating superior performance in their respective approaches. However, the fundamental limitation remains that each pairwise comparison provides at most 1 bit of information [20]. In addition, users struggle to choose between two similar trajectories [7]. In contrast, our method of comparative language feedback offers more informative input after observing just one trajectory, while maintaining intuitiveness and ease of use.

Robot Learning with Human Language Feedback. Benefiting from advancements in natural language processing, works have leveraged language for adjusting robot trajectories [13, 28, 29, 30, 31, 32], fine-tuning language models [12], and reward shaping [33, 34]. For example, Shi et al. [31] use language-conditioned behavior cloning (LCBC) for corrective language commands and improving policies. Lynch et al. [35] describe an approach for real-time guidance from humans using natural language in order to achieve a goal. Cui et al. [36] introduce an approach to use human language feedback to correct robot manipulation in real-time via shared autonomy. Goyal et al. [33] utilize human language combined with past action sequences to generate rewards, and in a separate work, Goyal et al. [34] leverage natural language to map image observations to rewards. However, these works focus on using human language as the instruction or correction to the robot, none of them has explored learning the reward function of humans entirely from comparative language feedback, which offers benefits over policy learning with generalizability and explainability. Additionally, our method iteratively updates the learned reward function for better preference alignment, providing an advantage over one-shot corrections [13, 29].

3 Problem Definition

Having reviewed the related literature, we are now ready to formally define the problem. We model the robot as a decision making agent in a standard finite-horizon Markov decision pro-



(a) Model Architecture (b) Language-based Reward Learning

Figure 2: Overview of our approach. (a) Architecture of the model that learns a shared latent space between trajectories and comparative language feedback. (b) Comparative language-based reward learning.

cess (MDP). Each robot trajectory consists of state-action pairs for T time steps[1]: $\tau = \{(s_0, a_0), (s_1, a_1), ..., (s_{T-1}, a_{T-1})\}$. A reward function $R(s_t, a_t)$, which is only known by the human user, encodes human preferences regarding the task. The reward of a trajectory is defined as: $R(\tau) = \sum_{t=0}^{T-1} R(s_t, a_t)$. For each trajectory, the human may provide language feedback l that attempts to improve the trajectory (based on the reward function) in one aspect, e.g., speed, distance to objects, height of the robot's arm, etc. As an example, the human might say "move farther from the stove" (see Figure 1) if the robot's trajectory would be higher-reward in that way. Our goal is to develop a framework where we learn the reward function based on such feedback.

Unfortunately, this task is doomed without any information about how language feedback relates to the different aspects of the task. In this work, we learn this relation with an offline pretraining phase where we collect a dataset that consists of pairs of robot trajectories and a language label describing how the two trajectories differ. We use this dataset to learn encoders that map trajectories and language feedback onto a shared latent space. We leverage these encoders to learn the preferences (reward functions) of different users based on their language feedback.

In the next section, we detail our approach to learning the latent space and explain how this learned latent space is utilized for iterative trajectory improvements and reward learning.

4 Our Approach

As shown in Figure 2, our approach is composed of two stages: First, we learn a shared latent space where robot trajectories and human language feedback are aligned. Second, we leverage this learned latent space to (1) improve robot trajectory or (2) learn human preferences.

4.1 Learning the Shared Latent Space

To learn a shared latent space for trajectories and language feedback, we collect a dataset of (τ^A, τ^B, l) tuples, where τ^A and τ^B are a pair of trajectories and l is a language utterance that describes the difference between the two trajectories. Note that this language utterance does not necessarily align with the human's preferences about the robot: it just describes a difference between the trajectories — it is possible to have $l =$"move faster" if the robot moves faster in τ^B than τ^A even though users want the robot to move slowly.

After collecting such a dataset, we propose the model visualized in Figure 2 to learn the shared latent space between trajectories and language utterances. Similar to [37], we use a neural network to encode each state-action pair (s_t, a_t) from the pair of trajectories, τ^A and τ^B, to embeddings ϕ_t^A and ϕ_t^B, respectively. We want these embeddings to contain information about aspects of the trajectories that the human may care and give feedback about.

To achieve this, we align the difference between the embeddings ϕ^A and ϕ^B with embeddings ψ_l of the language feedback l by using the following loss function:

$$L_{\text{align}}(\tau^A, \tau^B, l) = -\log\left(\text{sigmoid}\left(\psi_l^\top(\phi^B - \phi^A)\right)\right) \quad (1)$$

[1]Our work trivially extends to the cases where trajectory length is not fixed.

where the sigmoid can be considered as the probability that the language utterance l is inputted for the pair of trajectories τ^A and τ^B. Intuitively, when $\phi^B - \phi^A$ and ψ_l align, i.e., have the same direction for fixed magnitudes, the loss is minimized. This alignment will enable us to acquire the embedding of an improved trajectory as $\phi^A + \psi_l$ when given an initial trajectory τ^A and comparative language feedback l.

Instead of training a language encoder from scratch, which would require a prohibitively large and diverse dataset of (τ^A, τ^B, l) tuples, we use a pretrained T5 model [38]. To align embeddings of trajectories with those of language feedback, we first freeze T5 model and train the trajectory encoder. Subsequently, we perform co-finetuning of both components.

Additionally, we incorporate a normalization term into the loss function to help balance the magnitude of the embeddings. This term constrains the norms of the trajectory embeddings to remain below 1 and the norm of the language embeddings to be close to 1:

$$L_{\text{norm}}(\tau^A, \tau^B, l) = a \cdot \big(\max\{\|\phi^A\| - 1, 0\} + \max\{\|\phi^B\| - 1, 0\}\big) + b \cdot \|\psi_l - 1\|^2 \tag{2}$$

where a and b are two hyperparameters.

Overall, the objective we use in the model training consists of two terms:

$$L(\tau^A, \tau^B, l) = L_{\text{align}}(\tau^A, \tau^B, l) + L_{\text{norm}}(\tau^A, \tau^B, l), \tag{3}$$

which we use to train the trajectory encoder and finetune T5. Training this architecture gives us the ability to encode any trajectory and language utterance in the same latent space. In the next subsection, we demonstrate how this latent space is useful for iteratively improving robot trajectories and for learning human preferences based on their language feedback.

4.2 Utilizing the Learned Latent Space

The shared latent space aligns robot trajectories with human language feedback. This alignment enables an intuitive understanding of user preferences. We will now explore two primary ways to leverage this learned latent space: first, to iteratively improve the robot's trajectory, and second, to accurately learn user preferences.

4.2.1 Iterative Trajectory Improvement

Firstly, we leverage the latent space to iteratively improve an initial suboptimal robot trajectory. We start with showing the initial trajectory τ^0 to the user and asking for language feedback l^0. Upon receiving human's language feedback, we use our trained encoders to compute the trajectory embedding ϕ^0 and language embedding ψ_{l^0}. We then find the *improved trajectory* τ^1 such that its difference with τ^0 best aligns with the human's language feedback based on cosine similarity:

$$\tau^1 = \underset{\tau'}{\operatorname{argmax}} \frac{\psi_{l^0}^\top (\phi' - \phi^0)}{\|\phi'\|_2 \cdot \|\phi^0\|_2} \tag{4}$$

We iteratively continue this process for N iterations to obtain $\tau^0, \tau^1, \ldots, \tau^N$. In this work, we search over a discrete, predefined set of trajectories to solve the optimization in Eq. (4) for computational efficiency. It is, however, possible to use reinforcement learning or model-predictive control algorithms to solve this optimization at the expense of increased computational cost.

4.2.2 Reward Learning from Comparative Language Feedback

In addition to improving trajectories, we also utilize the learned latent space to learn the user's preference, i.e., their reward function. Previous approaches ask users to label their preference from a pair of trajectories, which requires the users to watch two trajectories in order to give at most 1 bit of information. In our language-based reward learning approach, for each query i, we only show users one trajectory τ_i to collect language feedback l_i based on their preferences. Given one trajectory, we construct an improved trajectory $\hat{\tau}_i$: $\phi_{\hat{\tau}_i} = \phi_{\tau_i} + \psi_{l_i}$. Note that $\hat{\tau}_i$ is just an imaginary trajectory that maps to $\phi_{\tau_i} + \psi_{l_i}$ in the learned latent space. Then following the Bradley-Terry model [39], which is commonly used in preference-based learning, we model a preference predictor with the reward function r_ξ as:

$$P_\xi(\hat{\tau}_i \succ \tau_i) = \frac{\exp r_\xi(\phi_{\hat{\tau}_i})}{\exp r_\xi(\phi_{\hat{\tau}_i}) + \exp r_\xi(\phi_{\tau_i})} \tag{5}$$

where r_ξ is a neural network parameterized with ξ. The reward function r_ξ is trained by minimizing the following negative loglikelihood loss:

$$L_{\texttt{Explicit}} = -\frac{1}{n}\sum_{i=0}^{n-1} \log P_\xi(\hat{\tau}_i \succ \tau_i)\,. \tag{6}$$

We now make an important observation about the comparative language feedback. When the user tells the robot to move farther from the stove, it indeed indicates a preference between the original trajectory and the improved trajectory that moves farther from the stove. However, this feedback contains much more information than this comparison. The user could use any comparative language utterance to teach the robot, but they specifically selected one about the distance to the stove. This means, with high probability, the improvement the robot may get from this feedback is higher than that of any other comparative language feedback.

Mathematically, this indicates a preference for $\hat{\tau}_i$ over $\tilde{\tau}_i : \phi_{\tilde{\tau}_i} = \phi_{\tau_i} + \psi_{\tilde{l}_i}$ where $\tilde{l}_i$ is any language utterance other than l_i. Again we apply the Bradley-Terry model to utilize this implicit preference:

$$P_\xi(\hat{\tau}_i \succ \tilde{\tau}_i) = \frac{\exp r_\xi(\phi_{\hat{\tau}_i})}{\exp r_\xi(\phi_{\hat{\tau}_i}) + \exp r_\xi(\phi_{\tilde{\tau}_i})} \tag{7}$$

Based on this, we sample k language feedback from a set of pre-collected language utterances other than l_i, and minimize the following loss about the chosen language feedback:

$$L_{\texttt{Implicit}} = -\frac{1}{k}\sum_{j=0}^{k-1} \log P_\xi(\hat{\tau}_i \succ \tilde{\tau}_{i,j}) \tag{8}$$

Overall, the loss for reward learning from comparative language feedback is as follows:

$$L_{\texttt{Reward}} = L_{\texttt{Explicit}} + L_{\texttt{Implicit}} = -\frac{1}{n}\sum_{i=0}^{n-1}\Big(\log P_\xi(\hat{\tau}_i \succ \tau_i) + \frac{1}{k}\sum_{j=0}^{k-1} \log P_\xi(\hat{\tau}_i \succ \tilde{\tau}_{i,j})\Big). \tag{9}$$

Training the reward function with this loss enables us to efficiently capture human preferences through comparative language feedback.

5 Experiments

We conducted simulation experiments and human subject studies to validate our methods with diverse tasks and features that humans may give feedback about.

5.1 Simulation Experiments

Synthetic humans. For simulation experiments, we synthesize the comparative language feedback from a simulated human with a simplified reward function R:

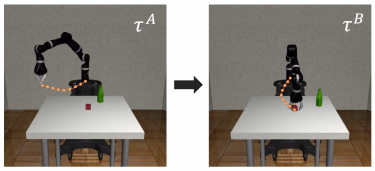
$$R(s_t, a_t) = w^\top \theta(s_t, a_t) \tag{10}$$

where θ is a function that maps a state-action pair to a vector of high-level features (e.g., speed, distance to the stove). We similarly define $\theta(\tau)$ to denote the sum of features over a trajectory τ. w is a vector of weights that maps the features to a scalar reward value. Both θ and w are unknown to our algorithm.

Given a true reward function w, let τ^* be the optimal trajectory under reward w. Then, the synthetic human gives the noisy language feedback l^0 when shown trajectory τ^0:

$$l^0 \sim \ell\left(\text{softmax}(w \odot (\theta_l^* - \theta_l^0))\right) \tag{11}$$

where θ^* and θ^0 denote the true cumulative features of τ^* and τ^0 respectively, $\odot$ denotes element-wise multiplication. The function ℓ takes a sample from the output of the softmax, which is computed across different features we designed for the simulated humans (see Appendix A), and outputs a language feedback that corresponds to the sampled feature. The language feedback to output is chosen randomly from all language utterances of that feature in the GPT-augmented dataset (see Appendix E).



"Move closer to the cube"

Figure 3: Each dataset sample is a pair of trajectories and a language feedback.

Environments. We experimented in two simulation environments: Robosuite [40] and Meta-World [41]. Robosuite has a Jaco robot arm at a table with a cube and a bottle, and the task is to pick up the cube. The states are given as 640×480 RGB images (resized to 224×224) and 25-dimensional

proprioception. The action space is 4-dimensional (the end-effector always points down). Meta-World has a Sawyer robot arm at a table, and the task is to push a button on the side of the table. The states are given as 480×320 RGB images (resized to 224×224) and 20-dimensional proprioception. The action space is 4-dimensional (the end-effector always points down).

Learning the Latent Space. Lists of language feedback were created for hand-crafted features (see Appendix A) to indicate a change (e.g., {'Move higher', 'Move lower'} for height). They were augmented with GPT 3.5 [42] to 629 sentences for Robosuite and 660 for Meta-World. The splits are 480, 74, 75 and 492, 84, 84 for the training, validation, and test sets. Note these features are only for synthetic dataset creation — training our architecture does not require hand-designed features.

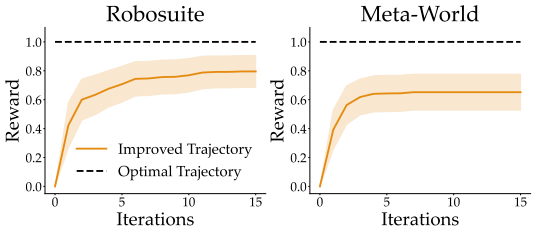


Figure 4: Results of experiments where we use simulated human language feedback to iteratively improve a robot trajectory (mean $\pm$ std over 100 runs). The dashed line represents *average* reward of optimal trajectories.

We trained RL policies with randomized weights w using stratified sampling [43] over the features, then generated rollout trajectories with timesteps $T = 500$ for each environment. For Robosuite, we generated 448 unique rollouts and 324 for Meta-World. The splits were 359, 44, 45 and 260, 32, 32 for training, validation, and test sets. Trajectories were paired within each set and matched with a language feedback that describes the change in each feature from τ^A to τ^B (see Figure 3).

We train the encoders with the loss in Eq. (3), and use accuracy as the metric to evaluate:

$$\text{Acc} = \frac{|\{(\tau^A, \tau^B, l) \mid \psi_l^\top(\phi^B - \phi^A) > 0\}|}{\text{Total number of samples}} . \tag{12}$$

Training the encoders with the training set of trajectories and language utterances, we observed a test accuracy of 84.9% in Robosuite and 82.9% in Meta-World. This indicates the trajectory and language embeddings are well-aligned. We found co-finetuning both trajectory encoder and language model, versus utilizing a frozen language model and only training the trajectory encoder, helps map trajectories and language utterances to the same latent space (see Table 2 in the Appendix).

Iterative Trajectory Improvement. Next, we conducted iterative trajectory improvement experiments based on the method we presented in Section 4.2.1. We set the number of iteration steps $N = 15$ and repeat the full experiment over 100 random seeds. The true rewards of these trajectories are shown in Fig. 4. In both environments, we consistently improve the trajectories, which showcases the effectiveness of the learned latent space and the improvement algorithm. However, it can be noted that our algorithm cannot reach the performance of the optimal trajectory. This is because every improvement iteration is completely independent from the previous iterations and the robot may get stuck in a loop between good, but non-optimal, trajectories.

Our reward learning algorithm presented in Section 4.2.2, on the other hand, utilizes the entire history of human feedback, so it should not suffer from this problem. We will now demonstrate this.

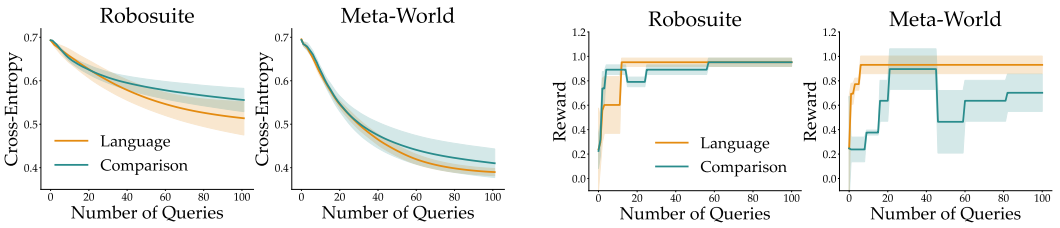
Reward Learning. We again randomly initialize the reward weights w to simulate human feedback. For each environment, we simulate three synthetic humans, and run the experiments with three random seeds for each simulated human. We use the loss in Eq. (8) to learn the reward function. We adopt cross-entropy $CE(P_w, P_\xi)$ as the evaluation metric, where

$$P_w(\tau^A \succ \tau^B) = \frac{\exp w^\top \theta(\tau^A)}{\exp w^\top \theta(\tau^A) + \exp w^\top \theta(\tau^B)} , \; P_\xi(\tau^A \succ \tau^B) = \frac{\exp r_\xi(\phi^A)}{\exp r_\xi(\phi^A) + \exp r_\xi(\phi^B)} \tag{13}$$

Another metric is the true reward value of the optimal trajectory selected from the test set based on the learned reward, reflecting how close the learned reward is to the true reward. All reward values are normalized between 0 and 1.

We compare our method against reward learning from comparisons, where for each query, the simulated user chooses a preferred trajectory from pair (τ^A, τ^B) and with probability shown in Eq. (13).

As indicated in Figure 5a, the cross-entropy of our method decreases faster than that of comparison preference learning in both environments, demonstrating that our approach converges more quickly than the baseline. Meanwhile, Figure 5b shows that the true reward of optimal trajectories reaches

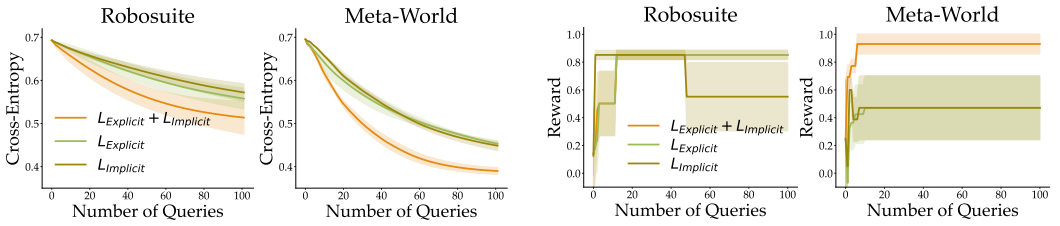


(a) Cross-Entropy (b) True Reward Values of Optimal Trajectory

Figure 5: Results of reward learning, averaged over 3 seeds. (a) Cross Entropy: our method converges faster. (b) True Reward of Optimal Trajectory: Our approach.

a value of 1 in a shorter time with our language-based method, especially in Meta-World, further supporting that it learns the reward function more efficiently.

Ablation Study. We conducted an ablation study to evaluate the different components of the loss function (Eq. (8)). As indicated in Figure 6a, the combination of both components outperforms using each component individually, demonstrating the validity of our loss function design. Also, using both components fits the true reward function better, as shown in Figure 6b.



(a) Cross-Entropy (b) True Reward Values of Optimal Trajectory

Figure 6: Results of Ablation Study. (a) Baseline Comparison: our method consistently outperforms the baseline. (b) Ablation Study: Using two components is better than applying each of them alone.

5.2 User Studies

To verify the effectiveness of our approach, we conducted human subject studies by recruiting 10 subjects (4 female, 6 male) from varying backgrounds and observing them to interact with our real robot setup. Our study has been approved by the IRB office of the University of Southern California.

Real Robot Setup. We used a WidowX 250 6DOF robot with the setup of [44] in front of a kitchen set (Figure 7). The task is to pick up a spoon while avoiding going over a pan on the stove. The states are given as 480×320 images resized to 224×224, and 22-dimensional proprioception. The actions are 7-dimensional (6DOF+gripper). To train the latent space, we collected a dataset of 321 trajectories with varying levels of success at picking up the spoon, avoiding the pan, and speed. These trajectories were divided into 192 for training, 64 for validation, and 65 for testing. 496 GPT-augmented sentences were split into 297 for training, 99 for validation, and 100 for testing. These sentences were paired with the trajectories in the same manner as in the simulation experiments. Another set of 32 trajectories was collected for the user studies following the pretraining phase.

Study 1: Trajectory Improvement. Users start with a suboptimal trajectory from the dataset and give language feedback for a maximum of 10 iterations until they are satisfied.

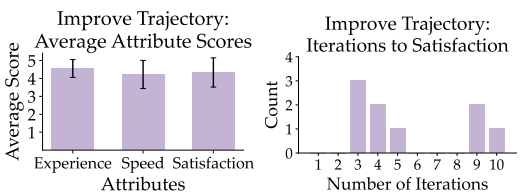


Figure 8: (Left) Average attribute scores. (Right) Iterations to satisfaction.

Study 2: Preference Learning. For 20 iterations, users give feedback (preference comparison or comparative language) on random trajectories and rate the optimal trajectory from the dataset with respect to the learned reward after every 5 iterations. To avoid bias, 5 users began with the comparison based method and 5 with the comparative language based method.

After each study, users were asked to complete post-study surveys (see Appendix B).

Figure 7: (Left) Closeup view of the kitchen set with the spoon hanging on the wall, above the pan on the stove. (Right) WidowX 250 6DOF robot arm, user, and text prompt for experiments.

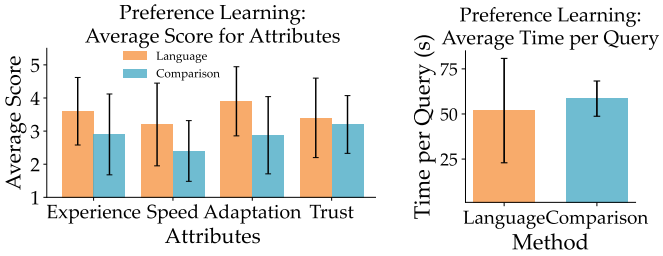


Figure 9: (Left) User ratings. (Right) Time per query.

Table 1: User satisfaction in aspects, not mutually exclusive between methods

	Satisfactory	
Method	Language	Comparison
Time Efficient	9 / 10	1 / 10
Convenient	6 / 10	8 / 10
Adaptable	8 / 10	3 / 10

Results. Our post-study surveys reveal that the first study has consistently positive responses for user experience and speed of adaptation, but has two peaks in iterations to satisfaction (Fig. 8). We conjecture that the dataset of 32 trajectories may not have contained trajectories that the users desired. As for the second study, our language-based method scored better than the comparison-based method **for all attributes**. The average score of the language-based method over all attributes is **23.9%** higher than the comparison-based method , illustrating the superior performance of our approach. In terms of time-efficiency, our method takes **11.3%** less time for users to answer each query as shown in Fig. 9, even though it includes the time it took for users to type their feedback. Indeed, users found the language method to be more time efficient and adaptable but found comparison method to be more convenient (Table 1). Fig. 10 shows the user rating of optimal trajectories given by the learned reward function. To quantitatively assess the learning efficiency, we follow prior work in active learning [9, 45, 46] by checking the area under curve (AUC) for both lines. We found that the AUC for the comparative language line is statistically significantly higher than that of preference comparison line ($p < 0.05$). This indicates that our language-based approach more efficiently captures human preferences compared to the comparison-based method.

An unexpected observation from Fig. 10 is that humans' ratings decreased from 5 queries to 10 queries while using comparative language feedback. This may be because of the relatively small subject size: it is possible that the average rating after 5 iterations, where the standard deviation is indeed large, is higher than it should be. Another potential explanation is that humans may be more lenient to the robot after 5 iterations than 10 iterations, especially in comparative language feedback where they only watch the robot 5 times to give their feedback as opposed to 10 times in preference comparisons.

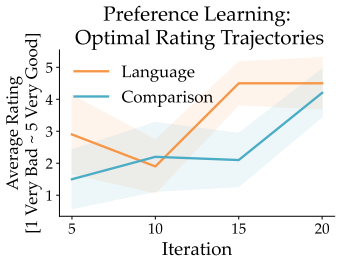


Figure 10: Average ratings of the optimal trajectory shown every 5 iterations for each method.

6 Conclusion

In this paper, we presented a robot learning framework to learn human preferences using comparative language feedback. For this, we aligned trajectories with language feedback in a shared latent space, then used it to improve trajectories and learn preferences. Experiments in simulation environments and real-world user studies suggest that our approach consistently outperforms comparison preference learning and is favored by most users for aspects such as time efficiency and adaptability.

Limitations and Future Work. Even though our user studies show some generalizability in the comparative language feedback, our model is limited by the feedback about the objects seen in the pretraining data. Additionally, we sample queries uniformly at random during reward learning. Based on these limitations, the future works include: (1) use image annotations to generate language feedback containing all objects, and (2) implement active learning to reduce the number of iterations required to reach the optimal trajectory.

Acknowledgments

This work was partially supported by NSF HCC grant #2310757 and a gift in support of the Center for Human-Compatible AI at UC Berkeley from the Open Philanthropy Foundation.

References

[1] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for autonomous cars that leverage effects on human actions. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2016. doi:10.15607/RSS.2016.XII.029.

[2] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[3] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[4] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. In *5th Annual Conference on Robot Learning*, 2021.

[5] D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. In *Proceedings of Robotics: Science and Systems (RSS)*, July 2017. doi:10.15607/RSS.2017.XIII.053.

[6] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[7] E. Bıyık, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, 41(1):45–67, 2022.

[8] N. Wilde, E. Biyik, D. Sadigh, and S. L. Smith. Learning reward functions from scale feedback. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[9] V. Myers, E. Biyik, N. Anari, and D. Sadigh. Learning multimodal rewards from rankings. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[10] A. Bajcsy, D. P. Losey, M. K. O'malley, and A. D. Dragan. Learning robot objectives from physical human interaction. In *Conference on Robot Learning*, pages 217–226. PMLR, 2017.

[11] A. Liang, J. Thomason, and E. Biyik. Visarl: Visual reinforcement learning guided by human saliency. In *ICRA2023 Workshop on Pretraining for Robotics (PT4R)*, 2023.

[12] J. A. Campos and J. Shern. Training language models with language feedback. In *ACL Workshop on Learning with Natural Language Supervision. 2022.*, 2022.

[13] P. Sharma, B. Sundaralingam, V. Blukis, C. Paxton, T. Hermans, A. Torralba, J. Andreas, and D. Fox. Correcting robot plans with natural language feedback. In *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[14] C. Wirth, R. Akrour, G. Neumann, J. Fürnkranz, et al. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

[15] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. In *International Conference on Machine Learning (ICML)*, 2024.

[16] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[17] E. Bıyık, D. A. Lazar, R. Pedarsani, and D. Sadigh. Incentivizing efficient equilibria in traffic networks with mixed autonomy. *IEEE Transactions on Control of Network Systems*, 8(4): 1717–1729, 2021.

[18] N. Dennler, D. Delgado, D. Zeng, S. Nikolaidis, and M. Matarić. The rosid tool: Empowering users to design multimodal signals for human-robot collaboration. In *18th International Symposium on Experimental Robotics (ISER)*, 2023.

[19] S. Casper, X. Davies, C. Shi, T. K. Gilbert, J. Scheurer, J. Rando, R. Freedman, T. Korbak, D. Lindner, P. Freire, T. Wang, S. Marks, C.-R. Segerie, M. Carroll, A. Peng, P. Christoffersen, M. Damani, S. Slocum, U. Anwar, A. Siththaranjan, M. Nadeau, E. J. Michaud, J. Pfau, D. Krasheninnikov, X. Chen, L. Langosco, P. Hase, E. Biyik, A. Dragan, D. Krueger, D. Sadigh, and D. Hadfield-Menell. Open problems and fundamental limitations of reinforcement learning from human feedback. *Transactions on Machine Learning Research (TMLR)*, 2024.

[20] E. Biyik, M. Palan, N. C. Landolfi, D. P. Losey, and D. Sadigh. Asking easy questions: A user-friendly approach to active reward learning. In *Proceedings of the 3rd Conference on Robot Learning (CoRL)*, 2019.

[21] C. Basu, M. Singhal, and A. D. Dragan. Learning from richer human guidance: Augmenting comparison-based learning with feature queries. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 132–140, 2018.

[22] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa. Expert intervention learning: An online framework for robot learning from explicit and implicit human feedback. *Autonomous Robots*, pages 1–15, 2022.

[23] P. Abbeel and A. Ng. Apprenticeship learning via inverse reinforcement learning. *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004*, 09 2004. doi:10.1007/978-0-387-30164-8_417.

[24] B. Akgün, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4:343 – 355, 2012. URL https://api.semanticscholar.org/CorpusID:10004846.

[25] E. Bıyık, N. Anari, and D. Sadigh. Batch active learning of reward functions from human preferences. *ACM Transactions on Human-Robot Interaction*.

[26] H. Sikchi, A. Saran, W. Goo, and S. Niekum. A ranking game for imitation learning. *Transactions on Machine Learning Research*, 2023.

[27] D. S. Brown, W. Goo, and S. Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pages 330–359. PMLR, 2020.

[28] A. Bucker, L. Figueredo, S. Haddadinl, A. Kapoor, S. Ma, and R. Bonatti. Reshaping robot trajectories using natural language commands: A study of multi-modal data alignment using transformers. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 978–984. IEEE, 2022.

[29] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, S. Vemprala, and R. Bonatti. Latte: Language trajectory transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7287–7294. IEEE, 2023.

[30] J. Yow, N. P. Garg, M. Ramanathan, W. T. Ang, et al. Extract–explainable trajectory corrections from language inputs using textual description of features. *arXiv preprint arXiv:2401.03701*, 2024.

[31] L. X. Shi, Z. Hu, T. Z. Zhao, A. Sharma, K. Pertsch, J. Luo, S. Levine, and C. Finn. Yell at your robot: Improving on-the-fly from language corrections, 2024.

[32] M. Han, Y. Zhu, S.-C. Zhu, Y. N. Wu, and Y. Zhu. Interpret: Interactive predicate learning from language feedback for generalizable task planning. *arXiv preprint arXiv:2405.19758*, 2024.

[33] P. Goyal, S. Niekum, and R. J. Mooney. Using natural language for reward shaping in reinforcement learning, 2019.

[34] P. Goyal, S. Niekum, and R. Mooney. Pixl2r: Guiding reinforcement learning using natural language by mapping pixels to rewards. In *Conference on Robot Learning*, pages 485–497. PMLR, 2021.

[35] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

[36] Y. Cui, S. Karamcheti, R. Palleti, N. Shivakumar, P. Liang, and D. Sadigh. No, to the right: Online language corrections for robotic manipulation via shared autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 93–101, 2023.

[37] S. M. Katz, A. Maleki, E. Bıyık, and M. J. Kochenderfer. Preference-based learning of reward function features. *arXiv preprint arXiv:2103.02727*, 2021.

[38] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[39] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[40] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[41] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[42] OpenAI. Gpt-3.5. Online. https://platform.openai.com/docs/models/gpt-3-5.

[43] M. J. Kochenderfer and T. A. Wheeler. *Algorithms for optimization*. Mit Press, 2019.

[44] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

[45] M. Culver, D. Kun, and S. Scott. Active learning to maximize area under the roc curve. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 149–158. IEEE, 2006.

[46] D. Fuchsgruber, T. Wollschläger, B. Charpentier, A. Oroz, and S. Günnemann. Uncertainty for active learning on graphs. *arXiv preprint arXiv:2405.01462*, 2024.

A Dataset Features

Robosuite Features

- The height of the robot's end-effector
- The speed of the end-effector
- The distance between the end-effector and the bottle
- The distance between the end-effector and the cube
- How well the robot lifts the cube (i.e., level of success of cube-lifting task)

The level of success of the cube-lifting task is quantified by: 1) whether or not the cube is lifted above the height of a success threshold or 2) a weighted sum of the distance to the cube and whether or not the end-effector is grasping the cube (the level of success in picking up the spoon).

Meta-World Features

- The height of the robot's end-effector.
- The velocity of the robot's end-effector.
- The distance between the end-effector and the button.

WidowX Features

- The velocity of the end-effector
- The distance between the end-effector and the pan
- How well the robot picks up the spoon (i.e. level of success)

The level of success is quantified by: 1) whether the spoon is grasped from the hook 2) a weighted sum of the distance to the spoon and whether the end-effector is grasping the spoon.

B User Study Surveys

Here are the questions in our user study surveys.

Pre-Study Survey. To assess the subjects' experience and level of skill with robotics and machine learning, we conducted a short pre-study survey before any of the experiments.

1. "Age"
2. "Gender"
3. "Race"
4. "Highest level of education"
5. "Please describe your level of robotics experience, on a scale from 1 to 5."
6. "Please describe your level of machine learning experience, on a scale from 1 to 5."
7. "Have you ever interacted with a robot before? Please describe:"

Post-Study Survey — Improve Trajectory. Users filled out this post-study survey after completing the Improve Trajectory experiment.

1. "Are you satisfied with the final trajectory?" [1 Completely unsatisfied - 5 Completely satisfied]
2. "How many iterations did it take for the robot to adapt to your feedback?"
3. "How fast does the robot adapt to your feedback?" [1 Very slow - 5 Very fast]
4. "How did the robot's learning capabilities affect your interaction experience?" [1 Negatively - 5 Positively]
5. "Overall, do you think the approach is effective?" [Y/N]
6. "Do you have any other comments?"

Post-Study Survey — Preference Learning. Users filled out this post-study survey after completing two methods of the preference learning experiment.

1. "What did you like about these methods? For each aspect below, please click the circle if you found the method satisfactory in that regard:" [Adaptability, Time efficiency, Convenience, None of them]
2. "Could you explain the reasons?"
3. "How long did it take for the robot to adapt to your feedback?" [1 Very slowly - 5 Very quickly]
4. "How did the robot's learning capabilities affect your interaction experience?" [1 Negatively - 5 Positively]
5. "Did you notice the robot learning or adapting to your behavior?" [1 Not at all - 5 Constantly]
6. "How did the robot's learning process affect your level of trust in its capabilities?" [1 Negatively - 5 Positively]
7. "Overall, which method do you prefer?" [Pair-wise comparison/Language feedback]
8. "Why do you prefer this method?"
9. "Is there anything you wish the system would do but currently does not?"
10. "Do you have any other comments?"

C Experiments — Additional Figures

C.1 The Effect of Co-finetuning

Co-finetuning both trajectory encoder and language model, versus utilizing a frozen language model and only training the trajectory encoder, helps map trajectories and language utterances to the same latent space. This is shown in Table 2.

Table 2: Test Accuracy of training with frozen language model and co-finetuning

Method	Robosuite	Metaworld
Co-finetune	0.8489	0.8288
Freeze T5	0.7625	0.7344

C.2 Response Time to Different Feedback Types

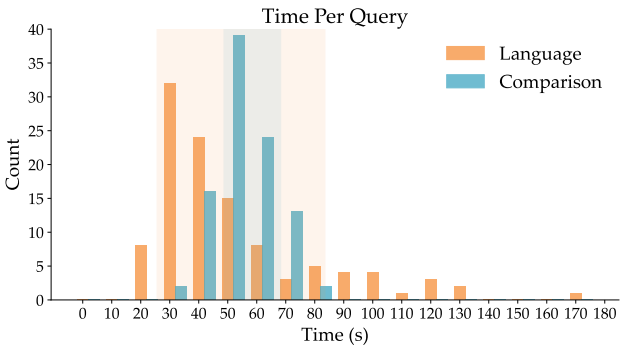


Figure 11: Time per query for both methods and averages, for 10 distinct users, indicated by color.

Figure 11 shows user-level differences of feedback length and time. The average of time of language-based reward learning is less than the comparison preference learning, illustrating that our method outperforms baseline in terms of the time-efficiency. However, the variance of language-based reward learning is larger than the baseline, which means users' thinking time varies a lot based on the trajectory shown.

D User Study Guidelines

Introduction. Welcome to our user study on human preference learning! In this study, you will interact with a robot and provide feedback based on your preferences. Please note that there will be no physical contact with the robot.

Task Description. The robot is situated in a kitchen setup and needs to pick up a spoon from the wall. However, there is a pan on the stove cooking something. For safety reasons, you should aim to have the robot successfully pick up the spoon while avoiding the pan. You can also adjust the robot's speed based on your preference.

Pre-study Survey. Before we begin, please complete a pre-study survey about your background and understanding of robotics and AI systems.

Experiment 1 - Improve Trajectory. In this experiment, the robot will first execute a suboptimal trajectory. Your task is to improve its behavior by providing comparative language feedback, such as "avoid the pan better." After each iteration, you will be asked if you are satisfied with the current trajectory. The experiment ends when you are satisfied with the trajectory or reach the maximum number of iterations. A post-study survey will be required after completing this experiment.

Experiment 2 - Preference Learning. In this experiment, you will be shown a series of queries and provide feedback on each one. Through this process, we will gradually learn your preferences and present you with the best trajectories based on the learned preferences. You will compare two approaches: language preference learning and pairwise preference learning.

1. Language Preference Learning
    - In each query, you will be shown one trajectory. You need to give comparative language feedback based on your preferences, such as "move faster," "avoid the pan better," or "be more adept at picking up the spoon." After every 5 queries, you will be shown the best trajectory based on the currently learned preferences and asked to rate it. You will complete a total of 20 queries.
2. Pairwise Preference Learning
    - In each query, you will be shown a pair of trajectories. You need to choose your preferred one. After every 5 queries, you will be shown the best trajectory based on the currently learned preferences and asked to rate it. You will complete a total of 20 queries.

After completing the experiment, you will be required to fill in a post-study survey.

E Original Language Feedback Utterances

All the original language feedback for each of the simulation environments are listed here. These sentences were then augmented with GPT-3.5.

Robosuite:

- Distance to the cube
    - Move farther from the cube.
    - Move further from the cube.
    - Move more distant from the cube.
    - Move less nearby from the cube.
    - Move nearer to the cube.
    - Move closer to the cube.
    - Move more nearby to the cube.
    - Move less distant to the cube.
- Distance to the bottle
    - Move further from the bottle.
    - Move farther from the bottle.

  - Move more distant from the bottle.
  - Move less nearby from the bottle.
  - Move nearer to the bottle.
  - Move closer to the bottle.
  - Move more nearby to the bottle.
  - Move less distant to the bottle.
- Height of the robot arm
  - Move taller.
  - Move at a greater height.
  - Move higher.
  - Move to a greater height.
  - Move lower.
  - Move at a lesser height.
  - Move shorter.
  - Move to a lower height.
- Speed of the robot arm
  - Move quicker.
  - Move swifter.
  - Move at a higher speed.
  - Move faster.
  - Move more quickly.
  - Move at a lower speed.
  - Move more moderate.
  - Move slower.
  - Move more sluggish.
  - Move more slowly.
- Proficiency at cube-lifting
  - Lift the cube better.
  - Lift the cube more successfully.
  - Lift the cube more effectively.
  - Lift the cube worse.
  - Lift the cube not as well.
  - Lift the cube less successfully.

Meta-World:

- Height of the robot arm
  - Move higher.
  - Move more up.
  - Move higher up from the table.
  - Increase the overall height of the trajectory.
  - Go higher up.
  - Move your gripper higher.
  - Don't go down, instead go up.
  - Stay higher and farther from the table.
  - Move your hand up as you perform the task.
  - Make sure to stay higher above the table, rather than lower.
  - Move lower.
  - Move more down.
  - Move lower to the table.

  - Decrease the overall height of the trajectory.
  - Go lower down.
  - Move your gripper lower.
  - Don’t go up, instead go down.
  - Stay lower and nearer to the table.
  - Move your hand lower as you perform the task.
  - Make sure to go lower to the table, rather than higher.
- Speed of the robot arm
  - Move faster.
  - Move at a quicker speed.
  - Increase the pace.
  - Press the button faster.
  - Increase your velocity.
  - Move your gripper faster.
  - Move to the button faster.
  - Don’t go too slowly, instead go quickly.
  - Move your hand faster as you perform the task.
  - Make sure to go much faster, rather than slower.
  - Move slower.
  - Move at a more sluggish speed.
  - Decrease the pace.
  - Press the button slower.
  - Decrease your velocity.
  - Move your gripper slower.
  - Move to the button slower.
  - Don’t go too fast, instead go more slowly.
  - Move your hand slower as you perform the task.
  - Make sure to go much slower, rather than faster.
- Distance to the button
  - Move farther from the button.
  - Increase distance from the button.
  - Stay farther from the button.
  - Give wider berth to the button.
  - Keep a larger distance from the button.
  - Keep your gripper farther away from the button.
  - Move your gripper away from the button.
  - Don’t go towards the button, instead move away from it.
  - Move your hand farther from the button on the wall as you perform the task.
  - Make sure to go much farther from the button, rather than closer to the button.
  - Move closer to the button.
  - Decrease distance from the button.
  - Stay closer to the button.
  - Get closer to the button.
  - Keep a smaller distance to the button.
  - Keep your gripper closer to the button.
  - Move your gripper so that it is closer to the button.
  - Don’t go away from the button, instead go towards it.
  - Move your hand closer to the button on the wall as you perform the task.
  - Make sure to go much closer to the button, rather than farther from the button.